

UUU	UUU	EEEEEEEEEEEEEE	TTTTTTTTTTTTTT	PPPPPPPPPPPPPP
UUU	UUU	EEEEEEEEEEEEEE	TTTTTTTTTTTTTT	PPPPPPPPPPPPPP
UUU	UUU	EEEEEEEEEEEEEE	TTTTTTTTTTTTTT	PPPPPPPPPPPPPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEEEEEEEEE	TTT	PPPPPPPPPPPPPP
UUU	UUU	EEEEEEEEEE	TTT	PPPPPPPPPPPPPP
UUU	UUU	EEEEEEEEEE	TTT	PPPPPPPPPPPPPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEE	TTT	PPP
UUU	UUU	EEE	TTT	PPP
UUUUUUUUUUUUUUU	UUUUUUUUUUUUU	EEEEEEEEEEEEEE	TTT	PPP
UUUUUUUUUUUUUUU	UUUUUUUUUUUUU	EEEEEEEEEEEEEE	TTT	PPP
UUUUUUUUUUUUUUU	UUUUUUUUUUUUU	EEEEEEEEEEEEEE	TTT	PPP

UU UU EEEEEEEEEE TTTTTTTTTT NN NN EEEEEEEEEE TTTTTTTTTT SSSSSSSS 000000 000000 000000
UU UU EEEEEEEEEE TTTTTTTTTT NN NN EEE TTTTTTTTTT SSSSSSSS 000000 000000 000000
UU UU EE TT NN NN EE TT SSS 00 00 00 00 00
UU UU EE TT NN NN EE TT SSS 00 00 00 00 00
UU UU EE NNNN NN EE TT SSS 00 00 00 00 00
UU UU EE NNNN NN EE TT SSS 00 00 00 00 00
UU UU EEEEEE TT NN NN EEEEEE TT SSSSS 00 00 00 00 00
UU UU EEEEEE TT NN NN EEEEEE TT SSSSS 00 00 00 00 00
UU UU EE TT NN NNNN EE TT SSS 0000 00 00 00 00
UU UU EE TT NN NNNN EE TT SSS 0000 00 00 00 00
UU UU EE TT NN NN EE TT SSS 00 00 00 00 00
UU UU EE TT NN NN EE TT SSS 00 00 00 00 00
UU UU EE TT NN NN EE TT SSS 00 00 00 00 00
UU UU EEEEEE TT NN NN EEEEEE TT SSSSSSS 000000 000000 000000
UU UU EEEEEE TT NN NN EEEEEE TT SSSSSSS 000000 000000 000000
LL IIIII SSSSSSS
LL IIIII SSSSSSS
LL II SS SSSSSSS
LL LLLLLLLL IIIII SSSSSSS
LL LLLLLLLL IIIII SSSSSSS

(2)	68	Declarations
(3)	127	Read-Only Data
(4)	231	Read/Write Data
(5)	361	RMS-32 Data Structures
(6)	388	Main Program
(9)	563	NICE_ROUTINE
(10)	750	System Service Exception Handler
(11)	889	RMS Error Handler
(12)	953	CTRL/C Handler
(13)	998	Error Exit
(14)	1054	Exit Handler

```
0000 1 .TITLE UETNETSO0 VAX/VMS UETP checker for DECnet counters
0000 2 .IDENT 'V04-000'
0000 3
0000 4 ****
0000 5 *
0000 6 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 7 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 8 * ALL RIGHTS RESERVED.
0000 9 *
0000 10 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 11 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 12 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 13 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 14 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 15 * TRANSFERRED.
0000 16 *
0000 17 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 18 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 19 * CORPORATION.
0000 20 *
0000 21 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 22 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 23 *
0000 24 *
0000 25 ****
0000 26
0000 27
0000 28 ++
0000 29 :FACILITY:
0000 30 This module will be distributed with VAX/VMS under the [SYSTEST]
0000 31 account.
0000 32
0000 33 :ABSTRACT:
0000 34 This program will report all errors indicating non-zero node and
0000 35 circuit counters for all nodes and circuits indicated in the
0000 36 UETININET.DAT file. If no counters indicate error then the node
0000 37 name and circuit name will be reported with a success message.
0000 38
0000 39 :ENVIRONMENT:
0000 40 This program will run in user access mode, with interrupts enabled
0000 41 at all times. This program requires the following privileges and
0000 42 quotas:
0000 43     NETMBX
0000 44
0000 45 --
0000 46
0000 47 :AUTHOR: Larry D. Jones.      CREATION DATE: November, 1981
0000 48
0000 49 :MODIFIED BY:
0000 50
0000 51     V03-004 RNH0002      Richard N. Holstein, 27-Mar-1983
0000 52     Make use of new UETP error messages. Turn off System Service
0000 53     failure exceptions when calling NML$INITIALIZE; we get snagged
0000 54     on a logical name that's used for debugging purposes only. Fix
0000 55     miscellaneous bugs in System Service error handling.
0000 56
0000 57     V03-003 RNH0001      Richard N. Holstein, 21-Nov-1983
```

0000 58 : Change error message wording when checking NICE messages.
0000 59 :
0000 60 : V03-002 LDJ0002 Larry D. Jones, 24-Jan-1983
0000 61 : Modified to conform to area network format.
0000 62 :
0000 63 : V03-001 LDJ0001 Larry D. Jones, 24-Dec-1981
0000 64 : Fixed missing node name access violation bug.
0000 65 :
0000 66 :**

```

0000 68      .SBttl Declarations
0000 69      :
0000 70      : INCLUDE FILES:
0000 71      :
0000 72      .LIBRARY /SHRLIBS:NMALIBRY.MLB/
0000 73      :
0000 74      :
0000 75      : MACROS:
0000 76      :
0000 77      $CHFDEF          : Condition handler frame definitions
0000 78      $DIBDEF          : Device information block definitions
0000 79      $NMADEF          : DECnet definitions
0000 80      $SHRDEF          : Shared messages
0000 81      $STSDEF          : Status return
0000 82      $UETPDEF          : UETP
0000 83      :
0000 84      .MACRO TBL_ENT ENT,VALUE,STRING
0000 85      .=PC1...
0000 86      .WORD <ENT@15>!VALUE
0000 87      .ADDRESS PC2...
0000 88      PC1...=.
0000 89      .=PC2...
0000 90      .ASCII /STRING/
0000 91      PC2...=.
0000 92      .ENDM TBL_ENT
0000 93      :
0000 94      : EQUATED SYMBOLS:
0000 95      :
0000 96      0000 : Facility number definitions:
0000 97      RMSSK_FACILITY = 1
0000 98      :
0000 99      0000 : SHR message definitions:
0000 100     UETP = UETPS_FACILITY@STSSV_FAC_NO ; Define the UETP facility code
0000 101     :
0000 102     UETPS_ABENDD = UETP!SHRS_ABENDD ; Define the UETP message codes
0000 103     UETPS_BEGIND = UETP!SHRS_BEGIND
0000 104     UETPS_ENDEDD = UETP!SHRS_ENDEDD
0000 105     UETPS_OPENIN = UETP!SHRS_OPENIN
0000 106     UETPS_TEXT = UETP!SHRS_TEXT
0000 107     :
0000 108     0000 : Internal flag bits...:
0000 109     SHRT_RPRTV = 0          : Set if short report format desired
0000 110     CONTROL_CV = 1         : Set if CTRL/C FST received
0000 111     CIR_CNT_BADV = 2        : Set if a bad circuit counter was detected
0000 112     NOD_CNT_BADV = 3        : Set if a bad node counter was detected
0000 113     BEGIN_MSGV = 6          : Set when "begin" msg has been output
0000 114     0000 : ...and corresponding masks:
0000 115     SHRT_RPRTM = 1@SHRT_RPRTV
0000 116     CONTROL_CM = 1@CONTROL_CV
0000 117     CIR_CNT_BADM = 1@CIR_CNT_BADV
0000 118     NOD_CNT_BADM = 1@NOD_CNT_BADV
0000 119     BEGIN_MSGM = 1@BEGIN_MSGV
0000 120     BIT7M = ^X80
0000 121     0000 : Miscellany:
0000 122     TEXT_BUFFER = 132       : Internal text buffer size
0000 123     NOD = 0                 : Node ID constant
0000 124     CIR = 1                 : Circuit ID constant

```

0000001A 0000 125 TBL_SIZE = 26 ; Network counter table size

0000 127 .SBTTL Read-Only Data
 00000000 128 .PSECT RODATA,NOEXE,NOWRT,PAGE
 0000 129
 0000 130 ACNT_NAME:
 53 45 54 53 59 53 00000008'010E0000' 0000 131 .ASCID /SYSTEST/ : Process name on exit
 54
 000E
 000F 132
 54 45 4E 54 45 55 00000017'010E0000' 000F 133 TSTNAM:
 30 30 53 001D 134 .ASCID /UETNETS00/ : This test name
 0020
 0020 135 NO_RMS_AST_TABLE:
 00000000' 0020 137 .LONG RMSS_BLN : List of errors for which...
 00000000' 0024 138 .LONG RMSS_BUSY : ...RMS cannot deliver an AST...
 00000000' 0028 139 .LONG RMSS_CDA : ...even if one has an ERR= arg
 00000000' 002C 140 .LONG RMSS_FAB : Note that we can search table...
 00000000' 0030 141 .LONG RMSS_RAB : ...via MATCHC since <31:16>...
 00000014 0034 142 NRAT_LENGTH = .-NO_RMS_AST_TABLE : ...pattern can't be in <15:0>
 0034
 0034 143
 45 44 4F 4D 0000003C'010E0000' 0034 144 MODE:
 0040 145 .ASCID /MODE/ : Run mode logical name
 0040 146
 0000 003F 0040 147 TTNAME_ROPTR:
 0000000A' 0044 148 .WORD 63,0
 0048 149 .ADDRESS TTNAME
 0048 150
 47 4F 4C 2E 0048 151 LOGEXT:
 004C 152 .ASCII /.LOG/ : Log file extention
 004C 153
 004C 154
 004C 155 CNTRLMSG:
 65 74 72 6F 62 41 00000054'010E0000' 004C 156 .ASCID \Aborted via a user CTRL/C\
 72 65 73 75 20 61 20 61 69 76 20 64 005A
 43 2F 4C 52 54 43 20 0066
 006D
 006D 157
 006D 158
 65 6C 69 66 00000075'010E0000' 006D 159 FILE:
 0079 160 .ASCID /file/ : Fills in RMS_ERR_STRING
 0079 161
 64 72 6F 63 65 72 00000081'010E0000' 0079 162 RECORD:
 0087 163 .ASCID /record/ : Fills in RMS_ERR_STRING
 0087 164
 41 21 20 53 4D 52 0000008F'010E0000' 0087 165 RMS_ERR_STRING:
 66 20 6E 69 20 72 6F 72 72 65 20 53 0095 : Announces an RMS error
 44 41 21 20 65 6C 69 00A1
 00A8
 00A8 166 .ASCID /RMS !AS error in file !AD/ :
 00A8 167
 00A8 168 NMLIMIT_ERR:
 00E3 169 .ASCID /Error during network communications initialization./
 00E3
 00E3 170
 72 20 45 43 49 4E 000000EB'010E0000' 00E3 171 ERR_MSG_CTR:
 6F 72 72 65 20 65 73 6E 6F 70 73 65 00F1 .ASCID /NICE response error code !XB, error message: !AC./

20 2C 42 58 21 20 65 64 6F 63 20 72 00FD
67 61 73 73 65 6D 20 72 6F 72 72 65 0109
2E 43 41 21 20 3A 65 0115
011C 173
011C 174 COUNTER_MSG:
011C 175 .ASCID /!AC !AC !AC !AC !AC = !UL./
41 21 20 43 41 21 00000124'010E0000' 011C
41 21 20 43 41 21 20 43 41 21 20 43 012A
2E 4C 55 21 20 3D 20 43 0136
013E
013E 176
013E 177 NODE:
013E 178 .ASCIIC /Node /
0144 179
0144 180 CIRCUIT:
0144 181 .ASCIIC /circuit/
014C 182
014C 183 TO:
014C 184 .ASCIIC /to/
014F 185
014F 186 THRU:
014F 187 .ASCIIC /over/
0154 188
0154 189 CASE FAILED:
0154 190 .ASCID /Unrecognized counter in NICE message./
6F 63 65 72 6E 55 0000015C'010E0000' 0154
74 6E 75 6F 63 20 64 65 7A 69 6E 67 0162
6D 20 45 43 49 4E 20 6E 69 20 72 65 016E
2E 65 67 61 73 73 65 017A
0181
0181 191
0181 192 CIRCUIT_OK:
0181 193 .ASCID /circuit !AC to !AC OK./
019F 194
019F 195 ZERO:
01A3 196 .LONG 0
01A3 197 CNTR_TBL:
01A3 198 PC1... =
01A3 199 .=.♦<IBL_SIZE*6>
023F 200 TBL_END:
023F 201 PC2... =
023F 202 .LIST MEB
023F 203 TBL_ENT CIR,NMASC_CTCIR_ACL,<arriving congestion loss>
023F 000001A3 023F
023F 8322 01A3
023F 0000023F 01A5
023F 0000023F 01A9
6F 63 20 67 6E 69 76 69 72 72 61 00' 023F
73 6F 6C 20 6E 6F 69 74 73 65 67 6E 024B
73 0257
18 023F
0258 204 .NLIST MEB
0258 205 TBL_ENT CIR,NMASC_CTCIR_CRL,<corruption loss>
0268 206 TBL_ENT CIR,NMASC_CTCIR_TCL,<transit congestion loss>
0280 207 TBL_ENT CIR,NMASC_CTCIR_LDN,<line down>

028A	208	TBL_ENT CIR,NMASC_CTCIR_IFL,<initialization failure>
02A1	209	TBL_ENT CIR,NMASC_CTCIR_DEI,<data errors inbound>
02B5	210	TBL_ENT CIR,NMASC_CTCIR_DEO,<data errors outbound>
02CA	211	TBL_ENT CIR,NMASC_CTCIR_RRT,<remote reply timeouts>
02E0	212	TBL_ENT CIR,NMASC_CTCIR_LRT,<local reply timeouts>
02F5	213	TBL_ENT CIR,NMASC_CTCIR_RBE,<remote buffer errors>
030A	214	TBL_ENT CIR,NMASC_CTCIR_LBE,<local buffer errors>
031E	215	TBL_ENT CIR,NMASC_CTCIR_SLT,<selection timeouts>
0331	216	TBL_ENT CIR,NMASC_CTCIR_RPE,<remote process errors>
0347	217	TBL_ENT CIR,NMASC_CTCIR_LPE,<local process errors>
035C	218	TBL_ENT CIR,NMASC_CTCIR_LIR,<locally initiated resets>
0375	219	TBL_ENT CIR,NMASC_CTCIR_RIR,<remotely initiated resets>
038F	220	TBL_ENT CIR,NMASC_CTCIR_NIR,<network initiated resets>
03A8	221	TBL_ENT NOD,NMASC_CTNOD_RTO,<response timeouts>
038A	222	TBL_ENT NOD,NMASC_CTNOD_RSE,<received connect resource errors>
03DB	223	TBL_ENT NOD,NMASC_CTNOD_APL,<aged packet loss>
03EC	224	TBL_ENT NOD,NMASC_CTNOD_NUL,<node unreachable packet loss>
0409	225	TBL_ENT NOD,NMASC_CTNOD_NOL,<node out of range packet loss>
0427	226	TBL_ENT NOD,NMASC_CTNOD_OPL,<oversized packet loss>
043D	227	TBL_ENT NOD,NMASC_CTNOD_PFE,<packet format error>
0451	228	TBL_ENT NOD,NMASC_CTNOD_RUL,<partial routing update loss>
0460	229	TBL_ENT NOD,NMASC_CTNOD_VER,<verification reject>

```

0481 231 .SBTTL Read/Write Data
00000000 232 .PSECT ?WDATA,WRT,NOEXE,PAGE
0000 233
0000 234 TTCHAN:
0000 235 .WORD 0 ; Channel associated with ctrl. term.
0000 236
0002 237 TTNAME_RWPTR:
0000 000B' 0002 238 .WORD TTNAME_LEN,0
0000000A' 0006 239 .ADDRESS TTNAME
0000 000A 240 TTNAME:
0000 241 .ASCII /SYSS$COMMAND/
0000000B 0015 242 TTNAME_LEN = .-TTNAME
00000049 0015 243 .BLKB 63-TTNAME_LEN
0049 244
0049 245
0049 246 FLAG: .WORD 0 ; Miscellaneous flag bits
0000 247 ; (See Equated Symbols for definitions)
004B 248
004B 249 DEV: .LONG DIB$K_LENGTH ; Device Information Block
00000074 004B 250 .ADDRESS DEVBUF
00000053' 004F 251 .BLKB DIB$K_LENGTH
000000C7 0053 252 DEVBUF:
00C7 253 .WORD TEXT_BUFFER,0 ; FAO output string descriptor
0000 0084, 00C7 254 .ADDRESS BUFFER
000000D7' 00CB 255 .WORD TEXT_BUFFER,0 ; Fake .ASCID buffer for misc. strings
0000 0084, 00CF 256 .ADDRESS BUFFER ; A word for length, a word for desc.
000000D7' COD3 257 .BLKB TEXT_BUFFER ; FAO output and other misc. buffer
00D7 258
00D7 259 BUFFER_PTR:
0000 0084, 00CF 260 .WORD TEXT_BUFFER,0 ; Cumulative error count at runtime
000000D7' 00D7 261 .ADDRESS BUFFER
0000015B 015B 262
015B 263 BUFFER: .BLKB TEXT_BUFFER ; Status value on program exit
00000000 015B 264
015F 265
015F 266 ERROR_COUNT: .LONG 0 ; Status value on program exit
00000000 015F 267
0163 268
0163 269 STATUS: .LONG 0 ; Status value on program exit
00000000 0163 270
0163 271
0163 272
00000167 0163 273 MSG_BLOCK: .BLKB 4 ; Auxiliary SGETMSG info
0167 274
0167 275
0167 276 EXIT_DESC: .LONG 0 ; Exit handler descriptor
00000000 0167 277
0000071C' 016B 278 .ADDRESS EXIT_HANDLER
00000001 016F 279 .LONG 1
0000015F' 0173 280 .ADDRESS STATUS
0177 281
0177 282 ARG_COUNT: .LONG 0 ; Argument counter used by ERROR_EXIT
00000000 0177 283
0178 284
0178 285 AREA_ADR_DESC: .LONG 0 ; Area descriptor
00000000, 017B 286 .ADDRESS 0
00000000' 017F 287

```

0183 288
00000000 0183 289 NODE_ADR_DESC:
00000000 0183 290 :LONG 0
0187 291 :ADDRESS 0
0188 292
00000005 0188 293 NICE_MSG:
0000019E 018F 294 :LONG NICE_SIZE
0193 295 :ADDRESS NICE_MESSAGE
0193 296
00000002 0193 297 NICE1_MSG:
000001A3 0197 298 :LONG NICE1_SIZE
0198 299 :ADDRESS NICET_MESSAGE
0198 300
0198 301 .ALIGN LONG
019C 302
0000 019C 303 AREA_WRD: ; Network area number
019C 304 .WORD 0
019E 305
019E 306 :
019E 307 : *** Warning ***
019E 308 : The following section of data must remain contiguous.
019E 309 :
019E 310 : NICE packets used to get the counters.
019E 311 :
019E 312 :
019E 313 NICE_MESSAGE:
14 019E 314 .BYTE NMASC_FNC_REA ; Read information function code
30 019F 315 .BYTE NMASC_OPINF_COUANMASV_OPT_INF ; OPTION = Node, Counters, Volatile
00 01A0 316 .BYTE NMASC_ENT_NOD ; Node format = node address
01A1 317 NODE_WRD:
0000 01A1 318 .WORD 0
00000005 01A3 319 NICE_SIZE = .-NICE_MESSAGE
01A3 320
01A3 321 NICE1_MESSAGE:
14 01A3 322 .BYTE NMASC_FNC_REA ; Read information function code
33 01A4 323 .BYTE <<NMASC_OPINF_COUANMASV_OPT_INF>!-
00000002 01A5 324 <NMASC_ENT_CIR> ; OPTION = Circuit, Counters, Volatile
01A5 325 NICE1_SIZE = .-NICET_MESSAGE
01A5 326
000001AF 01A5 327 CIRC_NAME:
01AF 328 .BLKB 10
01AF 329
01AF 330 :
01AF 331 : *** End of warning ***
01AF 332 :
01AF 333 :
000001B6 01AF 334 NODE_NAME:
01B6 335 .BLKB 7
000001B9 01B6 336
01B6 337 AREA_ADR:
01B9 338 .BLKB 3
01B9 339
000001BE 01B9 340 NODE_ADR:
01BE 341 .BLKB 5
01BE 342
000001D7 01BE 343 NAME:
01BE 344 .BLKB 25

00000000	01D7	345	
	01D7	346	COUNTER:
	01D7	347	.LONG 0
	01DB	348	
00000000	01DB	349	TYPE:
	01DB	350	.LONG 0
	01DF	351	
00000000	01DF	352	TYPE1:
	01DF	353	.LONG 0
	01E3	354	
00000000	01E3	355	TYPE2:
	01E3	356	.LONG 0
	01E7	357	
00000000	01E7	358	END_ADDR:
	01E7	359	.LONG 0

```
01EB 361      .SBTTL RMS-32 Data Structures
01EB 362      .ALIGN LONG
01EC 363
01EC 364  INI_FAB:                                ; Allocate FAB for UETININET
01EC 365      $FAB-
01EC 366      FAC = GET,-
01EC 367      RAT = CR,-
01EC 368      SHR = GEF,-
01EC 369      FNM = <UETININET.DAT>
023C 370
023C 371  INI_RAB:                                ; Allocate RAB for UETININET
023C 372      $RAB-
023C 373      FAB = INI_FAB,-
023C 374      UBF = BUFFER,-
023C 375      USZ = TEXT_BUFFER,-
023C 376      RBF = BUFFER
0280 377
0280 378
0280 379  LOG_FAB:                                ; Log file FAB
0280 380      $FAB      FNM = <UETNETS00.LOG>,-
0280 381      RAT = CR,-
0280 382      FAC = PUT
02D0 383  LOG_RAB:                                ; Log file RAB
02D0 384      $RAB      FAB = LOG_FAB,-
02D0 385      RBF = BUFFER,-
02D0 386      RSZ = TEXT_BUFFER
```

```

0314 388 .SBTTL Main Program
00000000 389 .PSECT UETNETS00,EXE,NOWRT,PAGE
0000 390
0000 391
0000 392
0000 393 .ENTRY UETNETS00,^M<> ; Entry mask
0002 394
60 04F7'CF DE 0002 395 MOVAL SSERROR (FP) ; Declare exception handler
000 396 $SETSFM_S ENBFLG = #1 ; Enable system service failure mode
0010 397 $DCLEXH_S DESBLK = EXIT_DESC ; Declare an exit handler
001B 398 $CREATE_FAB = LOG_FAB,-
001B 399 ERR = RMS_ERROR ; Create the log file
002A 400 $CONNECT_RAB = LOG_RAB,-
002A 401 ERR = RMS_ERROR ; Connect the RAB
0039 402 $OPEN_FAB =INI_FAB,-
0039 403 ERR = RMS_ERROR ; Open the UETININET.DAT file
0048 404 $CONNECT_RAB =INI_RAB,-
0048 405 ERR = RMS_ERROR ; Connect the RAB
0057 406 CLRL -(SP) ; Set the time stamp flag
000F'CF 7E D4 0059 407 PUSHAL TSTNAM ; Set the test name
02 DD 005D 408 PUSHL #2 ; Push the argument count
00741039 8F DD 005F 409 PUSHL #UETPS BEGIN!STSSK_SUCCESS ; Set the message code
00000000'GF 04 FB 0065 410 CALLS #4,G^LIB$SIGNAL ; Print the startup message
0049'CF 0040 8F A8 006C 411 BISW2 #BEGIN MSGM,FLAG ; Set flag so we don't type it twice
0073 412 $SETPRN_S PRCNAM = TSTNAM ; Set the process name
007E 413 10$: $TRNLOG_S LOGNAM = TTNAME_RWPTR,-
007E 414 RSLLEN = TTNAME_RWPTR,-
007E 415 RSLBUF = TTNAME_RWPTR,-
0006'CF 000A'CF DE 0097 416 MOVAL TTNAME_TTNAME_RWPTR+4 ; Translate the logical name
00000000'8F 50 D1 009E 417 CMPL R0,#SS$_NOTRAN ; Undo possible previous PPF fixup
13 13 00A5 418 BEQL 20$ ; Have we reached the end yet?
000A'CF 1B 91 00A7 419 CMPB #^X1B,TTNAME ; Br if yes
00 12 00AC 420 BNEQ 10$ ; Is this a process permanent file?
0002'CF 04 A2 00AE 421 SUBW #4,TTNAME_RWPTR ; Br if not
0006'CF 04 C0 00B3 422 ADDL #4,TTNAME_RWPTR+4 ; Remove RMS overhead from PPF name...
C4 11 0088 423 BRB 10$ ; ...
008A 424 20$: Now it's safe to retranslate
008A 425 $GETDEV_S DEVNAM = TTNAME_RWPTR,-
008A 426 PRIBUF = DEV ; Get its device type
00'8F 0057'CF 91 00CF 427 CMPB DEVBUF+DIB$B_DEVCLASS,ND$CS TERM ; Is this a terminal?
45 12 00D5 428 BNEQ 30$ ; BR if no
00D7 429 $ASSIGN_S DEVNAM = TTNAME_RWPTR,- ; Set up for CTRL/C AST's
00D7 430 CHAN = TTCHAN
00E8 431 $SQIOW_S CHAN = TTCHAN,- ; Enable CTRL/C AST's...
00E8 432 FUNC = #IOS_SETMODE!IOSM_CTRLCAST,-
00E8 433 P1 = CCASTHAND
00E8 434 PUSHAL TSTNAM ; ...and tell the user...
0COF'CF DF 0109 435 PUSHL #1
01 DD 010D 436 PUSHL #UETPS ABORT!STSSK_SUCCESS ; ...how to abort gracefully...
0074832B 8F DD 010F 437 CALLS #3,G^LIB$SIGNAL ; ...
00000000'GF 03 FB 0115 438 30$: $SETSFM_S ENBFLG = #0 ; While initializing net comm stuff...
011C 439 00000000'GF 00 FB 0125 440 CALLS #0,G^NMLSINITIALIZE
50 DD 012C 441 PUSHL R0
012E 442 $SETSFM_S ENBFLG = #1 ; ...don't die for lack of logical names
015F'CF 8ED0 0137 443 POPL STATUS

```

11 015F'CF E8 013C 445 BLBS STATUS_LOOP ; BR if we initialized correctly
00A8'CF DF 0141 446 PUSHAL NMLINIT_ERR
01 DD 0145 447 PUSHL #1
00741132 8F DD 0147 448 PUSHL #UETPS_TEXT!STS\$K_ERROR
04 DD 014D 449 PUSHL #4
0561 31 014F 450 BRW ERROR_EXIT
0152 451
0152 452 : Fall into main processing loop.

H 15

```

000D7'CF 20444E45 8F D1 0152 454 LOOP:
03 12 0161 455
0185 31 016A 456
          016C 457
          016F 458
          10$: 460 10$:
          025E'CF 20 3A 016F 461
          00D7'CF 0174 462
          01A5'CF 90 0177 463
          57 51 01 0184 464
          00D7'CF 56 28 0188 465
          59 025E'CF 56 A3 0190 466
          59 59 3C 0196 467
          59 59 B7 0199 468
          67 59 2E 3A 019B 469
          36 13 019F 470
          56 51 57 C3 01A1 471
          59 56 C2 01A5 472
          59 59 D7 01A8 473
          01B6'CF 56 90 01AA 474
          58 51 00 01AF 475
          01B7'CF 67 56 28 01B2 476
          017B'CF 56 98 01B8 477
          017F'CF 67 DE 01BD 478
          02 DD 01C2 479
          019C'CF DF 01C4 480
          017B'CF DF 01C8 481
00000000'GF 03 FB 01CC 482
          57 01 C1 01D3 483
          01D7 484 20$:
          67 59 20 3A 01D7 485
          56 51 57 C3 01D8 486
          01B9'CF 56 90 01DF 487
          58 51 00 01E4 488
          01BA'CF 67 56 28 01E7 489
          0183'CF 56 98 G ED 490
          0187'CF 67 DE 01F2 491
          02 DD 01F7 492
          01A1'CF DF 01F9 493
          0183'CF DF 01FD 494
          00000000'GF 03 FB 0201 495
          06 0A 019C'CF F0 0208 496
          58 000000D7'8F C3 0211 497
          01AF'CF 94 0219 498
          56 025E'CF 56 A3 021D 499
          17 13 0223 500
          01AF'CF 56 90 0225 501
          01AF'CF 97 022A 502
          56 97 022E 503
          58 51 01 C1 0230 504
          01B0'CF 68 56 28 0234 505
          30 11 023A 506
          023C 507 25$:
          53 01B0'CF 58 D4 023C 508
          DE 023E 510
          0152 455
          0152 456
          0161 457
          016A 458
          016C 459
          016F 460
          10$:
          3A 016F 461
          0174 462
          C3 0177 463
          90 017F 464
          C1 0184 465
          56 28 0188 466
          A3 0190 467
          3C 0196 468
          B7 0199 469
          3A 019B 470
          13 019F 471
          C3 01A1 472
          C2 01A5 473
          D7 01A8 474
          90 01AA 475
          00 01AF 476
          28 01B2 477
          98 01B8 478
          DE 01BD 479
          DD 01C2 480
          DF 01C4 481
          DF 01C8 482
          FB 01CC 483
          C1 01D3 484
          01D7 485
          3A 01D7 486
          C3 01D8 487
          90 01DF 488
          D0 01E4 489
          28 01E7 490
          G ED 491
          DE 01F2 492
          DD 01F7 493
          DF 01F9 494
          DF 01FD 495
          FB 0201 496
          F0 0208 497
          C3 0211 498
          94 0219 499
          A3 021D 500
          13 0223 501
          90 0225 502
          022A 503
          022E 504
          11 023A 505
          28 0234 506
          11 023A 507
          023C 508 25$:
          D4 023C 509
          DE 023E 510
          0152 455
          0152 456
          0161 457
          016A 458
          016C 459
          016F 460
          10$:
          3A 016F 461
          0174 462
          C3 0177 463
          90 017F 464
          C1 0184 465
          56 28 0188 466
          A3 0190 467
          3C 0196 468
          B7 0199 469
          3A 019B 470
          13 019F 471
          C3 01A1 472
          C2 01A5 473
          D7 01A8 474
          90 01AA 475
          00 01AF 476
          28 01B2 477
          98 01B8 478
          DE 01BD 479
          DD 01C2 480
          DF 01C4 481
          DF 01C8 482
          FB 01CC 483
          C1 01D3 484
          01D7 485
          3A 01D7 486
          C3 01D8 487
          90 01DF 488
          D0 01E4 489
          28 01E7 490
          G ED 491
          DE 01F2 492
          DD 01F7 493
          DF 01F9 494
          DF 01FD 495
          FB 0201 496
          F0 0208 497
          C3 0211 498
          94 0219 499
          A3 021D 500
          13 0223 501
          90 0225 502
          022A 503
          022E 504
          11 023A 505
          28 0234 506
          11 023A 507
          023C 508 25$:
          D4 023C 509
          DE 023E 510
          0152 455
          0152 456
          0161 457
          016A 458
          016C 459
          016F 460
          10$:
          3A 016F 461
          0174 462
          C3 0177 463
          90 017F 464
          C1 0184 465
          56 28 0188 466
          A3 0190 467
          3C 0196 468
          B7 0199 469
          3A 019B 470
          13 019F 471
          C3 01A1 472
          C2 01A5 473
          D7 01A8 474
          90 01AA 475
          00 01AF 476
          28 01B2 477
          98 01B8 478
          DE 01BD 479
          DD 01C2 480
          DF 01C4 481
          DF 01C8 482
          FB 01CC 483
          C1 01D3 484
          01D7 485
          3A 01D7 486
          C3 01D8 487
          90 01DF 488
          D0 01E4 489
          28 01E7 490
          G ED 491
          DE 01F2 492
          DD 01F7 493
          DF 01F9 494
          DF 01FD 495
          FB 0201 496
          F0 0208 497
          C3 0211 498
          94 0219 499
          A3 021D 500
          13 0223 501
          90 0225 502
          022A 503
          022E 504
          11 023A 505
          28 0234 506
          11 023A 507
          023C 508 25$:
          D4 023C 509
          DE 023E 510
          0152 455
          0152 456
          0161 457
          016A 458
          016C 459
          016F 460
          10$:
          3A 016F 461
          0174 462
          C3 0177 463
          90 017F 464
          C1 0184 465
          56 28 0188 466
          A3 0190 467
          3C 0196 468
          B7 0199 469
          3A 019B 470
          13 019F 471
          C3 01A1 472
          C2 01A5 473
          D7 01A8 474
          90 01AA 475
          00 01AF 476
          28 01B2 477
          98 01B8 478
          DE 01BD 479
          DD 01C2 480
          DF 01C4 481
          DF 01C8 482
          FB 01CC 483
          C1 01D3 484
          01D7 485
          3A 01D7 486
          C3 01D8 487
          90 01DF 488
          D0 01E4 489
          28 01E7 490
          G ED 491
          DE 01F2 492
          DD 01F7 493
          DF 01F9 494
          DF 01FD 495
          FB 0201 496
          F0 0208 497
          C3 0211 498
          94 0219 499
          A3 021D 500
          13 0223 501
          90 0225 502
          022A 503
          022E 504
          11 023A 505
          28 0234 506
          11 023A 507
          023C 508 25$:
          D4 023C 509
          DE 023E 510
          0152 455
          0152 456
          0161 457
          016A 458
          016C 459
          016F 460
          10$:
          3A 016F 461
          0174 462
          C3 0177 463
          90 017F 464
          C1 0184 465
          56 28 0188 466
          A3 0190 467
          3C 0196 468
          B7 0199 469
          3A 019B 470
          13 019F 471
          C3 01A1 472
          C2 01A5 473
          D7 01A8 474
          90 01AA 475
          00 01AF 476
          28 01B2 477
          98 01B8 478
          DE 01BD 479
          DD 01C2 480
          DF 01C4 481
          DF 01C8 482
          FB 01CC 483
          C1 01D3 484
          01D7 485
          3A 01D7 486
          C3 01D8 487
          90 01DF 488
          D0 01E4 489
          28 01E7 490
          G ED 491
          DE 01F2 492
          DD 01F7 493
          DF 01F9 494
          DF 01FD 495
          FB 0201 496
          F0 0208 497
          C3 0211 498
          94 0219 499
          A3 021D 500
          13 0223 501
          90 0225 502
          022A 503
          022E 504
          11 023A 505
          28 0234 506
          11 023A 507
          023C 508 25$:
          D4 023C 509
          DE 023E 510
          0152 455
          0152 456
          0161 457
          016A 458
          016C 459
          016F 460
          10$:
          3A 016F 461
          0174 462
          C3 0177 463
          90 017F 464
          C1 0184 465
          56 28 0188 466
          A3 0190 467
          3C 0196 468
          B7 0199 469
          3A 019B 470
          13 019F 471
          C3 01A1 472
          C2 01A5 473
          D7 01A8 474
          90 01AA 475
          00 01AF 476
          28 01B2 477
          98 01B8 478
          DE 01BD 479
          DD 01C2 480
          DF 01C4 481
          DF 01C8 482
          FB 01CC 483
          C1 01D3 484
          01D7 485
          3A 01D7 486
          C3 01D8 487
          90 01DF 488
          D0 01E4 489
          28 01E7 490
          G ED 491
          DE 01F2 492
          DD 01F7 493
          DF 01F9 494
          DF 01FD 495
          FB 0201 496
          F0 0208 497
          C3 0211 498
          94 0219 499
          A3 021D 500
          13 0223 501
          90 0225 502
          022A 503
          022E 504
          11 023A 505
          28 0234 506
          11 023A 507
          023C 508 25$:
          D4 023C 509
          DE 023E 510
          0152 455
          0152 456
          0161 457
          016A 458
          016C 459
          016F 460
          10$:
          3A 016F 461
          0174 462
          C3 0177 463
          90 017F 464
          C1 0184 465
          56 28 0188 466
          A3 0190 467
          3C 0196 468
          B7 0199 469
          3A 019B 470
          13 019F 471
          C3 01A1 472
          C2 01A5 473
          D7 01A8 474
          90 01AA 475
          00 01AF 476
          28 01B2 477
          98 01B8 478
          DE 01BD 479
          DD 01C2 480
          DF 01C4 481
          DF 01C8 482
          FB 01CC 483
          C1 01D3 484
          01D7 485
          3A 01D7 486
          C3 01D8 487
          90 01DF 488
          D0 01E4 489
          28 01E7 490
          G ED 491
          DE 01F2 492
          DD 01F7 493
          DF 01F9 494
          DF 01FD 495
          FB 0201 496
          F0 0208 497
          C3 0211 498
          94 0219 499
          A3 021D 500
          13 0223 501
          90 0225 502
          022A 503
          022E 504
          11 023A 505
          28 0234 506
          11 023A 507
          023C 508 25$:
          D4 023C 509
          DE 023E 510
          0152 455
          0152 456
          0161 457
          016A 458
          016C 459
          016F 460
          10$:
          3A 016F 461
          0174 462
          C3 0177 463
          90 017F 464
          C1 0184 465
          56 28 0188 466
          A3 0190 467
          3C 0196 468
          B7 0199 469
          3A 019B 470
          13 019F 471
          C3 01A1 472
          C2 01A5 473
          D7 01A8 474
          90 01AA 475
          00 01AF 476
          28 01B2 477
          98 01B8 478
          DE 01BD 479
          DD 01C2 480
          DF 01C4 481
          DF 01C8 482
          FB 01CC 483
          C1 01D3 484
          01D7 485
          3A 01D7 486
          C3 01D8 487
          90 01DF 488
          D0 01E4 489
          28 01E7 490
          G ED 491
          DE 01F2 492
          DD 01F7 493
          DF 01F9 494
          DF 01FD 495
          FB 0201 496
          F0 0208 497
          C3 0211 498
          94 0219 499
          A3 021D 500
          13 0223 501
          90 0225 502
          022A 503
          022E 504
          11 023A 505
          28 0234 506
          11 023A 507
          023C 508 25$:
          D4 023C 509
          DE 023E 510
          0152 455
          0152 456
          0161 457
          016A 458
          016C 459
          016F 460
          10$:
          3A 016F 461
          0174 462
          C3 0177 463
          90 017F 464
          C1 0184 465
          56 28 0188 466
          A3 0190 467
          3C 0196 468
          B7 0199 469
          3A 019B 470
          13 019F 471
          C3 01A1 472
          C2 01A5 473
          D7 01A8 474
          90 01AA 475
          00 01AF 476
          28 01B2 477
          98 01B8 478
          DE 01BD 479
          DD 01C2 480
          DF 01C4 481
          DF 01C8 482
          FB 01CC 483
          C1 01D3 484
          01D7 485
          3A 01D7 486
          C3 01D8 487
          90 01DF 488
          D0 01E4 489
          28 01E7 490
          G ED 491
          DE 01F2 492
          DD 01F7 493
          DF 01F9 494
          DF 01FD 495
          FB 0201 496
          F0 0208 497
          C3 0211 498
          94 0219 499
          A3 021D 500
          13 0223 501
          90 0225 502
          022A 503
          022E 504
          11 023A 505
          28 02
```

5B	017B'CF 5B 63	017B'CF 5B 2F	95 11 9A 2C	0243 0247 0249 0250	511 512 513 515	TSTB BEQL MOVZBL INCL MOVC5	AREA_ADR_DESC 27\$ AREA_ADR_DESC,R11 R11 AREA_ADR_DESC,AREA_ADR+1,- #^A/.,(R3)	; Is there an area number? ; BR if not ; Get area string length ; Add one for the area decimal point ; Save area number and decimal point
01BA'CF	0183'CF 01AF'CF 63	0183'CF 5B 28	C0 90 025A 0257 0264 026B	025A 518 519 520 521 522	518 519 520 521 522	27\$: ADDL2 MOVB MOVC3	NODE_ADR_DESC,R11 R11,NODE_NAME NODE_ADR_DESC,NODE_ADR+1,- (R3)	; Add node number size ; Store the size ; Save node number
00000000'GF 05 0049'CF 0049'CF	0324'CF 018B'CF 02	0324'CF 018B'CF 02	DF DF FB	026C 0270 0274	523 524 525	30\$: PUSHAL PUSHAL CALLS	NICE_ROUTINE NICE_MSG #2,G^NML\$PROCESS_NICE	; Get the node counters
0193'CF	0193'CF 01A5'CF 0324'CF 0193'CF	0193'CF 01A5'CF 0324'CF 0193'CF	D0 80 DF DF	0286 028B 0292 0296	529 530 531 532	40\$: MOVL ADDB2 PUSHAL PUSHAL CALLS	#NICE1_SIZE+1,NICE1_MSG CIRC_NAME,NICE1_MSG NICE_ROUTINE NICET_MSG #2,G^NML\$PROCESS_NICE	; Calculate NICE packet size ; Add in circuit name size ; Get the circuit counters
01DB'CF	00000000'GF 4A 0049'CF 44 0049'CF 01AF'CF 01AF'CF 07	00000000'GF 4A 0049'CF 44 0049'CF 01AF'CF 01AF'CF 07	02 02 02 02 02	02A1 02A7 02AD 02B4 02B8	533 534 535 536 538	BBSC BBSC MOVAL TSTB BNEQ	#NOD_CNT_BADV,FLAG,60\$ #CIR_CNT_BADV,FLAG,60\$ NODE_NAME,TYPE NODE_NAME 50\$; BR if counters found bad ; BR if counters found bad ; Save the node name address ; Anything there? ; BR if yes else...
01DB'CF	01B9'CF	01B9'CF	DE	028A	539	50\$: MOVAL	NODE_ADR,TYPE	; ...use the node address
			02C1	02C1	540	SFAO_S	CTRSTR = CIRCUIT OK,- OUTLEN = BUFFER PTR,- OUTBUF = FAO BUF,- P1 = #NAME,- P2 = TYPE	; Print the circuit OK message
			02C1	02C1	541	PUSHAL	BUFFER_PTR	; Push the string address
			02C1	02C1	542	PUSHL	#1	; Push the parameter counter
			02C1	02C1	543	PUSHL	#UETPS_TEXT!STS\$K_SUCCESS	; Push signal name
			02C1	02C1	544	CALLS	#3,G^LIB\$SIGNAL	; Print circuit OK
			02C1	02C1	545	BRW	LOOP	; Do the next record
			02DE	02E2	546			
			02E4	02EA	547			
			02F1	02F1	548			
			31	02F1	549			
			FE5E		550	60\$:		
					551			

00000000'GF 00 02F4 553 SUC_EXIT:
00 00 FB 02F4 554 CALLS #0,G^NML\$TERMINATE ; Terminate the NML session
00 00 DD 02FB 555 PUSHL #0 ; Set the time flag
000F'CF DF 02FD 556 PUSHAL TSTNAM ; Push the test name
02 02 DD 0301 557 PUSHL #2 ; Push arg count
00741081 8F DD 0303 558 PUSHL #UETPS ENDEDD!STSSK_SUCCESS ; Push signal name
00000000'GF 04 FB 0309 559 CALLS #4,G^LIB\$SIGNAL ; Output the message
015F'CF 10000000'8F DD 0310 560 MOVL #SS\$ NORMAL!STSSM_INHIB_MSG,STATUS ; Set successful exit status
0319 561 SEXIT_S STATUS ; Exit with the status

0324 563 .SBTTL NICE_ROUTINE
 0324 564 ++
 0324 565 : FUNCTIONAL DESCRIPTION:
 0324 566 This routine is the NICE response servicing routine. All calls to
 0324 567 NML\$PROCESS_NICE specify this routine as the action routine.
 0324 568
 0324 569 : CALLING SEQUENCE:
 0324 570 PUSHAL NICE_ROUTINE
 0324 571 PUSHAL NICE_MSG_DESC
 0324 572 CALLS #2, G^NM\$PROCESS_NICE
 0324 573
 0324 574 : INPUT PARAMETERS:
 0324 575 4(AP) = Address of a response message descriptor
 0324 576
 0324 577 : IMPLICIT INPUTS:
 0324 578 NONE
 0324 579
 0324 580 : OUTPUT PARAMETERS:
 0324 581 NONE
 0324 582
 0324 583 : IMPLICIT OUTPUTS:
 0324 584 Error or success messages
 0324 585
 0324 586 : COMPLETION CODES:
 0324 587 NONE
 0324 588
 0324 589 : SIDE EFFECTS:
 0324 590 NONE
 0324 591
 0324 592 :--
 0324 593
 0324 594 NICE_ROUTINE:
 0324 595 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
 01E7'CF 56 04 AC 0FFC 0324 596 MOVL 4(AP),R6 : Get the descriptor address
 04 57 66 3C 0326 597 MOVZWL (R6),R7 : Get the response size
 04 A6 57 C1 032A 598 ADDL3 R7,4(R6),END_ADR : Save the response end address
 56 04 A6 DO 0334 599 MOVL 4(R6),R6 : Get the response address
 58 86 9A 0338 600 MOVZBL (R6)+,R8 : Get the status code
 02 58 91 033B 601 CMPB R8,#NMASC_STS_MOR : If this is a more message then...
 03 12 033E 602 BNEQ 10\$
 01B3 31 0340 603 BRW NICE_EXIT : ...exit
 80 8F 58 91 0343 604 10\$: CMPB R8,#NMASC_STS_DON : If this is a done message then...
 03 12 0347 605 BNEQ 20\$
 01AA 31 0349 606 BRW NICE_EXIT : ...exit
 01 58 91 034C 607 20\$: CMPB R8,#NMASC_STS_SUC : If this is a success then...
 43 13 034F 608 BEQL CHECK_IT : ...process the response
 0351 609
 0351 610
 0351 611
 0351 612 : The nice response is in error and it is reported to the user.
 0351 613
 59 FF A6 9A 0351 614 MOVZBL -1(R6),R9 : Get error code
 5A 02 A6 DE 0355 615 MOVAL 2(R6),R10 : Get the error message address
 0359 616 \$FAO_S CTRSTR = ERR_MSG_CTR,-
 0359 617 OUTLEN = BUFFER_PTR,-
 0359 618 OUTBUF = FAO_BUF,-
 0359 619 P1 = R9,-

00CF'CF	DF	0359	620	PUSHAL	P2 = R10	
01	DD	0370	621	PUSHL	BUFFER_PTR	; Push the string address
00741132 8F	DD	0374	622	PUSHL	#1	; Push the argument count
00000000'GF	FB	0376	623	PUSHL	#UETPS_TEXT!STSSK_ERROR	; Push the signal name
0049'CF	03	037C	624	CALLS	#3 G^IB\$SIGNAL	; Print the error message
015F'CF	04	A8	0383	BISW2	#CIR_CNT_BADM,FLAG	; Set failure bit
00000000'8F	DO	0388	625	MOVL	\$\$\$\$_BADPARAM,STATUS	; Set return status failure code
0162	31	0391	626	BRW	NICE_EXIT	; Thats it
			627			
			628	CHECK_IT:		
56 03	CO	0394	629	ADDL2	#3,R6	; Skip error stuff
59 01A3'CF	DE	0397	630	MOVAL	CNTR_TBL,R9	; Set table address
0049'CF	04	AA	039C	BICW2	#CIR_CNT_BADM,FLAG	; Clear the bad flag
01A1'CF	66	B1	03A1	CMPW	(R6),NODE_WRD	; Is this a node response?
	03	12	03A6	BNEQ	10\$; BR if not...
56 02	CO	03A8	634	ADDL2	#2,R6	; ...else skip the node address word
57 66	9A	03AB	635	MOVZBL	(R6),R7	
57 80 8F	8A	03AE	636	BICB2	#BIT7M,R7	; Get the size of the name
01BE'CF	66 57	D6	03B2	INCL	R7	; Incase this is the executor node
56 57	28	03B4	638	MOVC3	R7,(R6),NAME	; Add in the count byte
	CO	03BA	639	ADDL2	R7,R6	; Save the name
			640			; Skip the name
01E7'CF	56	D1	03BD	CHK_LOOP:		
	03	12	03C2	CMPL	R6-END_ADDR	; All done?
012F	31	03C4	642	BNEQ	10\$; BR if not...
		03C7	643	BRW	NICE_EXIT	; ...else bail out
58 86	3C	03C7	645	10\$:		
00	EF	03CA	646	MOVZWL	(R6)+,R8	; Get cntrl desc
0C		03CC	647	EXTZV	#NMASV_CNT_TYP,-	
58 58		03CD	648		#NMASV_CNT_TYP,-	
		03CF	649		R8,R8	; Get the counter type
5A 89	3C	03CF	650	20\$:		
00	EF	03D2	651	MOVZWL	(R9)+,R10	; Get a table code
0C		03D4	652	EXTZV	#NMASV_CNT_TYP,-	
5B 5A		03D5	653		#NMASV_CNT_TYP,-	
5B 58	D1	03D7	654		R10,R11	; Get the counter type
48	13	05DA	655	CMPL	80\$; Is this it?
59 04	CO	03DC	656	BEQL	80\$; BR if yes
59	D1	03DF	657	ADDL2	#4,R9	; Skip name pointer
E7	12	03E6	658	CMPL	R9,#TBL_END	; End of table?
59 01A3'CF	DE	03E8	659	BNEQ	20\$; BR if not
0D	EF	03ED	660	MOVAL	CNTR_TBL,R9	; Set table address
02		03EF	661	EXTZV	#NMASV_CNT_WID,-	
58 FE	A6	03F0	662		#NMASV_CNT_WID,-	
03 FE A6	OC	E1	03F3		-2(R6),R8	; Get the counter width
56 02	CO	03F8	663	BBC	#NMASV_CNT_MAP,-2(R6),30\$; If not a mapped counter then carry on el
		03FB	664	ADDL2	#2,R6	; ...skip the map word
02 01 58	8F	03FB	665	30\$:		
		03FF	666	CASEB	R8,#1,#2	; Skip the counter
		0017'	03FF		.WORD 50\$-40\$	
		001C'	0401		.WORD 60\$-40\$	
		0022'	0403		.WORD 70\$-40\$	
0154'CF	DF	0405	672	PUSHAL	CASE_FAILED	; Push the string address
01	DD	0409	673	PUSHL	#1	; Push the argument count
00741132 8F	DD	040B	674	PUSHL	#UETPS_TEXT!STSSK_ERROR	; Push the signal name
03	DD	0411	675	PUSHL	#3	; Push the argument count
029D	31	0413	676	BRW	ERROR_EXIT	; Thats it

56	FFA2	D6	0416	677	50\$:	INCL	R6	CHK_LOOP	; Skip a byte counter
56	02	C0	0418	678		BRW			
56	FF9C	31	041B	679		ADDL2	#2,R6	CHK_LOOP	; Skip a word counter
56	04	C0	0421	680	60\$:	BRW			
56	FF96	31	0424	681		ADDL2	#4,R6	CHK_LOOP	; Skip a long word counter
03	58	FE	0427	682		BRW			
03	58	A6	0429	683	70\$:	EXTZV	#NMASV_CNT_WID,-		
03	58	A6	042A	684			#NMASV_CNT_WID,-		
03	58	0C	042D	685			-2(R6),R8		
02	01	58	0432	686	80\$:	BBC	#NMASV_CNT_MAP,-2(R6),90\$; Get the counter width	
02	01	58	0435	687		ADDL2	#2,R6	; If not a mapped counter then carry on el	
02	01	58	0439	688				; ...skip the map word	
02	01	58	0439	689	90\$:	CASEB	R8,#1,#2	; Skip the counter	
0017'	0439		0439	690					
001F'	043B		043B	691					
0027'	043D		043D	692	100\$:				
0154'CF	DF	043F	043F	693		PUSHAL	CASE_FAILED	; Push the string address	
00741132	8F	DD	0443	694		PUSHL	#1	; Push the argument count	
03	DD	0445	0445	695		PUSHL	#UETPS_TEXT!STSSK_ERROR	; Push the signal name	
0263	31	0448	0448	696		PUSHL	#3	; Push the argument count	
0263	31	044D	044D	697		BRW	ERROR_EXIT	; Thats it	
01D7'CF	86	9A	0450	700	110\$:	MOVZBL	(R6)+,COUNTER	; Get a byte counter	
000D	31	0455	0450	701		BRW	140\$		
01D7'CF	86	3C	0458	702	120\$:	MOVZWL	(R6)+,COUNTER	; Get a word counter	
0005	31	045D	0458	703		BRW	140\$		
01D7'CF	86	DD	0460	704	130\$:	MOVL	(R6)+,COUNTER	; Get a long word counter	
59	01A3'CF	08	0465	705		BNEQ	150\$		
59	01A3'CF	DE	0467	706		MOVAL	CNTR_TBL,R9	; Reset the table address	
59	FF4E	31	046C	707		BRW	CHK_LOOP	; BR if counter was zero	
0049'CF	04	A8	046F	708	140\$:	BISW2	#CIR_CNT_BADM,FLAG	; Set the bad one flag	
01DB'CF	013E'CF	DE	0474	709		MOVAL	NODE_TYPE	; Set the default entity type of node	
5A	01AF'CF	DE	047B	710		MOVAL	NODE_NAME,R10	; Save the node name address	
6A	95	0480	047B	711		TSTB	(R10)	; Anything there	
5A	01B9'CF	DE	0482	712		BNEQ	160\$; BR if yes else...	
5A	01B9'CF	DE	0484	713		MOVAL	NODE_ADDR,R10	; ...use the node address	
01DF'CF	014F'CF	DE	0489	714	150\$:	MOVAL	THRU,TYPE1	; Set for node THRU circuit format	
01E3'CF	01A5'CF	DE	0490	715		MOVAL	CIRC_NAME,TYPE2		
18	FF	A9	0497	716		BBC	#7,-T(R9),170\$; Check to see if we guessed right	
01DB'CF	0144'CF	DE	049C	717		MOVAL	CIRCUIT_TYPE	; If not set type to circuit	
01DF'CF	014C'CF	DE	04A3	718		MOVAL	TO,TYPE1	; Set up for circuit to node format	
01E3'CF	5A	DO	04AA	719		MOVL	R10,TYPE2		
5A	01BE'CF	DE	04AF	720		MOVAL	NAME,R10		
5A	01BE'CF	DE	0484	721	160\$:	\$FAO_S	CTRSTR = COUNTER_MSG,-	; Generate a bad counter message	
5A	01BE'CF	DE	0484	722			OUTLEN = BUFFER_PTR,-		
5A	01BE'CF	DE	0484	723			OUTBUF = FAO_BUF,-		
5A	01BE'CF	DE	0484	724	170\$:				

04B4	734	P1	= TYPE,-	
04B4	735	P2	= R10,-	
04B4	736	P3	= TYPE1,-	
04B4	737	P4	= TYPE2,-	
04B4	738	P5	= (R9),-	
04B4	739	P6	= COUNTER	
00CF'CF	DF	04DB	740	PUSHAL BUFFER_PTR ; Push the string address
01	DD	04DF	741	PUSHL #1 ; Push the argument count
00741133 8F	DD	04E1	742	PUSHL #UETPS_TEXT!STSSK_INFO ; Push the signal name
00000000'GF	FB	04E7	743	CALLS #3, G^IB\$SIGNAL ; Print the error message
59	01A3'CF	DE	04EE	744 MOVAL CNTR_TBL,R9 ; Reset the counter table pointer
FEC7	31	04F3	745	BRW CHK_COOP ; Thats it
		04F6	746	
		04F6	747	NICE_EXIT: RET
		04	04F6	748

BCDEFGHIJKLMNOPBCDEFGHIJKLMNOPBCDEFGHIJKLMNOPBCDEFGHI

04F7 750 .SBTTL System Service Exception Handler
 04F7 751 :++
 04F7 752 : FUNCTIONAL DESCRIPTION:
 04F7 753 : This routine is executed if a system service or RMS error occurs or
 04F7 754 : if a LIB\$SIGNAL system service is used to output a message.
 04F7 755 :
 04F7 756 : CALLING SEQUENCE:
 04F7 757 : Entered via an exception from the system
 04F7 758 :
 04F7 759 : INPUT PARAMETERS:
 04F7 760 : ERROR_COUNT = previous cumulative error count
 04F7 761 :
 04F7 762 : AP ----> 2
 04F7 763 :
 04F7 764 :
 04F7 765 :
 04F7 766 :
 04F7 767 :
 04F7 768 :
 04F7 769 :
 04F7 770 :
 04F7 771 :
 04F7 772 :
 04F7 773 :
 04F7 774 :
 04F7 775 :
 04F7 776 :
 04F7 777 :
 04F7 778 :
 04F7 779 :
 04F7 780 :
 04F7 781 :
 04F7 782 :
 04F7 783 :
 04F7 784 :
 04F7 785 :
 04F7 786 :
 04F7 787 :
 04F7 788 :
 04F7 789 : IMPLICIT INPUTS:
 04F7 790 : NONE
 04F7 791 :
 04F7 792 : OUTPUT PARAMETERS:
 04F7 793 : NONE
 04F7 794 :
 04F7 795 : IMPLICIT OUTPUTS:
 04F7 796 : NONE
 04F7 797 :
 04F7 798 : COMPLETION CODES:
 04F7 799 : NONE
 04F7 800 :
 04F7 801 : SIDE EFFECTS:
 04F7 802 : NONE
 04F7 803 :--
 04F7 804 :
 04F7 805 : SSERROR:
 OFFC 04F7 806 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; Entry mask

04F9 807
04F9 808
0502 809
0504 810
0507 811
0509 812
050B 813 10\$:
050B 814
0514 815
0516 816
0519 817
051B 818
051D 819 20\$:
051D 820
0521 821
0525 822
0527 823
0528 824
052E 825
0530 826
0533 827
0533 828
0533 829
0544 830
0546 831 30\$:
0546 832
054D 833
054F 834
0551 835
0552 836
0558 837
055A 838
0561 839
0565 840
0566 841
0569 842
056B 843 40\$:
056B 844
056D 845
0576 846
0578 847
0581 848
0584 849
0585 850 50\$:
0585 851
058A 852
058C 853
0593 854
0595 855
0595 856
0595 857
0595 858
0595 859
05AC 860
05B0 861
05B2 862
05B6 863

\$SETAST_S ENBFLG = #0
PUSHL #1
CMPL S^#SSS_WASSET, R0
BEQL 10\$
CLRL (SP)

\$SETSFM_S ENBFLG = #0
PUSHL #1
CMPL S^#SSS_WASSET, R0
BEQL 20\$
CLRL (SP)

MOVL CHF\$L_SIGARGLST(AP), R6
MOVQ CHF\$L_SIG_NAME(R6), R9
CMPZV #STS\$V_FAC_NO,-
#STS\$S_FAC_NO,-
R9 #UETPS_FACILITY
BNEQ 30\$
SUBL2 #2, CHF\$L_SIG_ARGS(R6)
\$PUTMSG_S MSGVEC = -
CHF\$L_SIG_ARGS(R6), -
ACTRTN = 80\$

BRB 40\$

CMPL #SSS_SSFAIL, R9
BNEQ 50\$
CMPZV #STS\$V_FAC_NO,-
#STS\$S_FAC_NO,-
R10, #RMSS_FACILITY
BNEQ 50\$
BICL2 #^XF0000000, R10
MATCHC #4, CHF\$L_SIG_ARGS(R6), -
#NRAT_LENGTH,-
NO RMS_AST_TABLE
BEQL 50\$

POPR #^M<R0>
\$SETSFM_S ENBFLG = R0
POPR #^M<R0>
\$SETAST_S ENBFLG = R0
MOVL S^#SSS_NORMAL, R0
RET

MOVL R9, STATUS
CLRL R8
CMPL #SSS_SSFAIL, R9
BNEQ 70\$
\$GETMSG_S MSGID = R10,-
MSGLEN = BUFFER_PTR,-
BUFADR = FAO_BUF,-
FLAGS = #14,-
OUTADR = MSG_BLOCK

TSTB MSG_BLOCK+1
BEQL 60\$
PUSHL BUFFER_PTR
PUSHL #1

; Disable AST delivery
; Assume ASTs were enabled
; Were ASTs enabled?
; BR if they were
; Set ASTs to remain disabled

; Disable SS failure mode
; Assume SS failure mode was enabled
; Was SS failure mode enabled?
; BR if it was
; Set SS failure mode to remain off

; Get the signal array pointer
; Get NAME in R9 and ARG1 in R10
; Is this a message from LIB\$SIGNAL?

; BR if this is not a UETP exception
; Drop the PC and PSL
; Print the message

; Restore ASTs and SS fail mode

; RMS failures are SysSvc failures
; BR if this can't be an RMS failure
; Is it an RMS failure?

; BR if not
; Strip control bits from status code
; Is it an RMS failure for which...

; ...no AST can be delivered?
; BR if so - must give error here

; Restore SS failure mode...

; Restore AST enable...

; Supply a standard status for exit
; Resume processing (or goto RMS_ERROR)

; Save the status
; Assume for now it's not SS failure
; But is it a System Service failure?
; BR if not - no special case message
; Get SS failure code associated text

; Get FAO arg count for SS failure code
; Don't use \$GETMSG if no \$FAO args...
; ...else build up..
; ...a message describing...

00741130	8F	DD	05B8	864		PUSHL	#UETPS_TEXT	; ...why the System Service failed	
00	5A	FO	05BE	865		INSV	R10,#STS\$V_SEVERITY-	; Give the message...	
6E	03		05C1	866			#STS\$S_SEVERITY,(SP)	; ...the correct severity code	
58	03	DO	05C3	867		MOVL	#3,R8	; Count the number of args we pushed	
	05	11	05C6	868		BRB	70\$		
			05C8	869	60\$:				
	5A	DD	05C8	870		PUSHL	R10	; Save SS failure code	
58	01	DO	05CA	871		MOVL	#1,R8	; Count the number of args we pushed	
			05CD	872	70\$:				
57	66	04	C5	05CD	873	MULL3	#4,CHF\$L_SIG_ARGS(R6),R7	; Get arglist length in bytes	
	5E	57	C2	05D1	874	SUBL2	R7,SP	; Save the current signal array...	
6E	04	A6	57	28	05D4	875	MOVC3	R7,CHF\$L_SIG_NAME(R6),(SP)	; ...on the stack
7E	66	58	C1	05D9	876	ADDL3	R8,CHF\$L_SIG_ARGS(R6),-(SP)	; Push the current arg count	
	00D3		31	05DD	877	BRW	ERROR_EXIT		
			05E0	878					
			05E0	879	80\$:				
		0004	05E0	880		.WORD	^M<R2>	; PUTMSG action routine	
52	04	AC	DO	05E2	881	MOVL	4(AP),R2	; Get the message descriptor address	
02F2'CF	62	3C	05E6	882		MOVZWL	(R2),LOG_RAB+RAB\$W_RSZ	; Get the message size	
02F8'CF	04	A2	DO	05EB	883	MOVL	4(R2),LOG_RAB+RAB\$E_RBF	; Set the message address	
			05F1	884		SPUT	RAB = LOG_RAB,-		
			05F1	885			ERR = RMS_ERROR	; Write the log file	
50	00000000'8F	DO	0600	886		MOVL	#SS\$NORMÄL,R0	; Set the return status code	
			0607	887		RET			

```

0608 889 .SBTTL RMS Error Handler
0608 890 ++
0608 891 :+ FUNCTIONAL DESCRIPTION:
0608 892 :+ This routine handles error returns from RMS calls.
0608 893 :
0608 894 :+ CALLING SEQUENCE:
0608 895 :+ Called by RMS when a file processing error is found.
0608 896 :
0608 897 :+ INPUT PARAMETERS:
0608 898 :+ NONE
0608 899 :
0608 900 :+ IMPLICIT INPUTS:
0608 901 :+ The FAB or RAB associated with the RMS call.
0608 902 :
0608 903 :+ OUTPUT PARAMETERS:
0608 904 :+ NONE
0608 905 :
0608 906 :+ IMPLICIT OUTPUTS:
0608 907 :+ Error message
0608 908 :
0608 909 :+ COMPLETION CODES:
0608 910 :+ NONE
0608 911 :
0608 912 :+ SIDE EFFECTS:
0608 913 :+ Program may exit, depending on severity of the error.
0608 914 :
0608 915 :-
0608 916 :
0608 917 RMS_ERROR:
0608 918 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; Entry mask
0608 919 :
060A 920 MOVL 4(AP),R6 ; See whether we're dealing with...
060A 921 CMPB #FAB$C_BID,FAB$B_BID(R6) ; ...a FAB or a RAB
060E 922 BNEQ 10$ ; BR if it's a RAB
0611 923 MOVAL FILE,R7 ; FAB-specific code: text string...
0613 924 MOVL R6,R8 ; ...address of FAB...
0618 925 PUSHL FAB$L-STV(R6) ; ...STV field for error...
061B 926 PUSHL FAB$L-STS(R6) ; ...STS field for error...
061E 927 MOVL FAB$L-STS(R6),STATUS ; ...and save the error code
0621 928 BRB RMS_COMMON ; FAB and RAB share other code
0627 929 10$:
0629 930 MOVAL RECORD,R7 ; RAB-specific code: text string...
0629 931 MOVL RAB$L-FAB(R6),R8 ; ...address of associated FAB...
062E 932 PUSHL RAB$L-STV(R6) ; ...STV field for error...
0632 933 PUSHL RAB$L-STS(R6) ; ...STS field for error...
0635 934 MOVL RAB$L-STS(R6),STATUS ; ...and save the error code
0638 935 RMS_COMMON:
063E 936 MOVZBL FAB$B_FNS(R8),R10 ; Get the file name size
0642 937 $FAO_S CTRSTR = RMS_ERR_STRING,- ; Common code, prepare error message...
0642 938 OUTLEN = BUFFER_PTR,-
0642 939 OUTBUF = FAO_BUF,-
0642 940 P1 = R7 =
0642 941 P2 = R10 =
0642 942 P3 = FAB$L_FNA(R8)
065C 943 PUSHAL BUFFER_PTR ; ...and arguments for ERROR_EXIT...
0660 944 PUSHL #1 ; ...
0662 945 PUSHL #UETPS_TEXT ; ...

```

59 015F'CF 00 EF 0668 946 EXTZV #STS\$V_SEVERITY,-
6E 59 03 066A 947 #STS\$S_SEVERITY,-
05 88 066B 948 STATUS R9 ; ...get the severity code...
003C 31 0672 950 BISB2 R9,(SP) ; ...and add it into the signal name
05 DD 0674 951 PUSHL #5 : Current arg count
BRW ERROR_EXIT

0677 953 .SBTTL CTRL/C Handler
0677 954 :++
0677 955 : FUNCTIONAL DESCRIPTION:
0677 956 : This routine handles CTRL/C AST's
0677 957
0677 958 : CALLING SEQUENCE:
0677 959 : Called via AST
0677 960
0677 961 : INPUT PARAMETERS:
0677 962 : NONE
0677 963
0677 964 : IMPLICIT INPUTS:
0677 965 : NONE
0677 966
0677 967 : OUTPUT PARAMETERS:
0677 968 : NONE
0677 969
0677 970 : IMPLICIT OUTPUTS:
0677 971 : NONE
0677 972
0677 973 : COMPLETION CODES:
0677 974 : NONE
0677 975
0677 976 : SIDE EFFECTS:
0677 977 : NONE
0677 978
0677 979 :--
0677 980
0677 981 CCASTHAND:
OFFC 0677 982 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; Entry mask
0679 983
004C'CF DF 0679 984 PUSHAL CNTRLMSG ; Set message pointer
01 DD 067D 985 PUSHL #1 ; Set arg count
00741130 8F DD 067F 986 PUSHL #UETPS_TEXT!STSSK_WARNING ; Set signal name
000F'CF DF 0685 987 PUSHL #0 ; Indicate an abnormal termination
000F'CF DF 0687 988 PUSHAL TSTNAM
02 DD 068B 989 PUSHL #2
007410E0 8F DD 068D 990 PUSHL #UETPS_ABEND!STSSK_WARNING ; ...
00000000'GF 07 FB 0693 991 CALLS #7,G^LIB\$SIGNAL ; Output the message
0049'CF 02 A8 069A 992 BISW2 #CONTROL CM,FLAG ; Set CTRL/C flag bit...
0049'CF 02 D0 069F 993 MOVL #<SSS CONTROLCB^C7- ; ...and exit status
015F'CF 10000000'8F 06A0 994 !STSSR_WARNING-
06A0 995 !STSSM_INHIB_MSG>,STATUS
06A8 996 \$EXIT_S STATUS ; Terminate program cleanly

06B3 998 .SBTTL Error Exit
 06B3 999 ++
 06B3 1000 : FUNCTIONAL DESCRIPTION:
 06B3 1001 : This routine prints an error message and exits.
 06B3 1002 :
 06B3 1003 : CALLING SEQUENCE:
 06B3 1004 : MOVx error status value,STATUS
 06B3 1005 : PUSHx error specific information on the stack
 06B3 1006 : PUSHL current argument count
 06B3 1007 : BRW ERROR_EXIT
 06B3 1008 :
 06B3 1009 : INPUT PARAMETERS:
 06B3 1010 : Arguments to LIB\$SIGNAL, as above
 06B3 1011 :
 06B3 1012 : IMPLICIT INPUTS:
 06B3 1013 : NONE
 06B3 1014 :
 06B3 1015 : OUTPUT PARAMETERS:
 06B3 1016 : Message to SYSSOUTPUT and SYS\$ERROR
 06B3 1017 :
 06B3 1018 : IMPLICIT OUTPUTS:
 06B3 1019 : Program exit
 06B3 1020 :
 06B3 1021 : COMPLETION CODES:
 06B3 1022 : NONE
 06B3 1023 :
 06B3 1024 : SIDE EFFECTS:
 06B3 1025 : NONE
 06B3 1026 :
 06B3 1027 :--
 06B3 1028 :
 06B3 1029 : ERROR_EXIT:
 06B3 1030 :
 15 0049'CF 06 E0 06BC 1031 : \$SETAST_S ENBFLG = #0 : ASTs can play havoc with messages
 7E D4 06C2 1032 : BBS #BEGIN_MSGV,FLAG,10\$: BR if 'begin' msg has already been output
 000F'CF DF 06C4 1033 : CLRL -(SP) : Set the time stamp flag
 02 DD 06C8 1034 : PUSHAL TSTNAM : Set the test name
 00741039 8F DD 06CA 1035 : PUSHL #2 : Push the argument count
 00000000'GF 04 FB 06D0 1036 : PUSHL #UETPS BEGIN!STS\$K_SUCCESS : Set the message code
 06D7 1037 : CALLS #4,G^LIB\$SIGNAL : Print the startup message
 0177'CF 08 8E C1 06D7 1038 10\$: ADDL3 (SP)+,#8,ARG_COUNT : Get total # args, pop partial count
 015B'CF D6 06DD 1040 : INCL ERROR_COUNT : Keep running error count
 00 DD 06E1 1041 : PUSHL #0 : Push the time parameter
 0C0F'CF F 06E3 1042 : PUSHAL TSTNAM : Push test name...
 02 DD 06E7 1043 : PUSHL #2 : ...arg count...
 007410E2 8F DD 06E9 1044 : PUSHL #UETPS ABEND!STS\$K_ERROR : ...and signal name
 015B'CF DD 06EF 1045 : PUSHL ERROR_COUNT : Finish off arg list...
 000F'CF DF 06F3 1046 : PUSHAL TSTNAM :
 02 DD 06F7 1047 : PUSHL #2 :
 00748022 8F DD 06F9 1048 : PUSHL #UETPS ERBOXPROC!STS\$K_ERROR : ...for error box message
 00000000'GF 0177'CF FB 06FF 1049 : CALLS ARG_COUNT,G^LIB\$SIGNAL : Truly hitch
 0708 1050 :
 0708 1051 :
 015F'CF 10000000 8F C8 0708 1052 : BISL #STS\$M_INHIB_MSG,STATUS : Don't print messages twice.
 0711 1052 : SEXIT_S STATUS : Exit in error

071C 1054 .SBTTL Exit Handler
071C 1055 :++
071C 1056 : FUNCTIONAL DESCRIPTION:
071C 1057 : This routine handles cleanup on exits.
071C 1058
071C 1059 : CALLING SEQUENCE:
071C 1060 : Invoked automatically by \$EXIT System Service.
071C 1061
071C 1062 : INPUT PARAMETERS:
071C 1063 : Location STATUS contains the exit status, FLAG has synchronizing bits.
071C 1064
071C 1065 : IMPLICIT INPUTS:
071C 1066 : NONE
071C 1067
071C 1068 : OUTPUT PARAMETERS:
071C 1069 : NONE
071C 1070
071C 1071 : IMPLICIT OUTPUTS:
071C 1072 : Various files are de-accessed, the process name is reset, and any
071C 1073 : necessary synchronization with UETPDEV01 is carried out.
071C 1074
071C 1075 : COMPLETION CODES:
071C 1076 : NONE
071C 1077
071C 1078 : SIDE EFFECTS:
071C 1079 : NONE
071C 1080
071C 1081 :--
071C 1082
071C 1083 EXIT_HANDLER:
OFFC 071C 1084 .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; Entry mask
071E 1085
071E 1086 \$SETSFN_S ENBFLG = #0 : Turn off System Service failure mode
0727 1087 \$SETAST_S ENBFLG = #0 : We're finished - no more ASTs
0730 1088 \$DISCONNECT RAB = INI_RAB : Disconnect the RAB from the FAB
073B 1089 \$CLOSE FAB = INI_FAB : Close the UETINIDEV.DAT file
0746 1090 \$DISCONNECT RAB = LOG_RAB : Disconnect the RAB from the FAB
0751 1091 \$CLOSE FAB = LOG_FAB : Close the UETNETS00.LOG file
04 075C 1092 \$SETPRN_S PRCNAM = ACNT_NAME : Reset the process name
0767 1093 RET : That's all folks!
0768 1094
0768 1095 .END UETNETS00

SS.TAB	= 000002D0	R	03	FAB\$L_FNA	= 0000002C
SS.TABEND	= 00000314	R	03	FAB\$L_FOP	= 00000004
SS.TMP	= 00000000			FAB\$L_STS	= 00000008
SS.TMP1	= 00000001			FAB\$L_STV	= 0000000C
SS.TMP2	= 000000CF			FAB\$V_CHAN_MODE	= 00000002
SS.TMPX	= 0000000D	R	04	FAB\$V_CR	= 00000001
SS.TMPX1	= 0000000D			FAB\$V_FILE_MODE	= 00000004
SST1	= 00000001			FAB\$V_GET	= 00000001
SST2	= 00000006			FAB\$V_LNM_MODE	= 00000000
ACNT_NAME	00000000	R	02	FAB\$V_PUT	= 00000000
AREA_ADR	000001B6	R	03	FAB\$W_GBC	= 00000048
AREA_ADR_DESC	0000017B	R	03	FAO_BUF	000000C7 R 03
AREA_WRD	0000C019C	R	03	FILE	00000060 R 02
ARG_COUNT	00000177	R	03	FLAG	00000049 R 03
BEGIN_MSGM	= 00000040			INI_FAB	000001EC R 03
BEGIN_MSGV	= 00000006			INI_RAB	0000023C R 03
BIT7M	= 00000080			IOSM_CTRLCAST	***** X 05
BUFFER	000000D7	R	03	IOS_SETMODE	***** X 05
BUFFER_PTR	000000CF	R	03	LIB\$SIGNAL	***** X 05
CASE_FAILED	00000154	R	02	LOGEXT	00000048 R 02
CCASTHAND	00000677	R	05	LOG_FAB	00000280 R 03
CHECK_IT	0000C0394	R	05	LOG_RAB	000002D0 R 03
CHF\$L_SIGARGLST	= 00000004			LOOP	00000152 R 05
CHF\$L_SIG_ARG1	= 00000008			MODE	00000034 R 02
CHF\$L_SIG_ARGS	= 00000000			MSG_BLOCK	00000163 R 03
CHF\$L_SIG_NAME	= 00000004			NAME	000001BE R 03
CHK_LOOP	= 000003BD	R	05	NICE1_MESSAGE	000001A3 R 03
CIR	= 00000001			NICE1_MSG	00000193 R 03
CIRCUIT	00000144	R	02	NICE1_SIZE	= 00000002
CIRCUIT_OK	00000181	R	02	NICE_EXIT	000004F6 R 05
CIRC_NAME	000001A5	R	03	NICE_MESSAGE	0000019E R 03
CIR_CNT_BADM	= 00000004			NICE_MSG	00000188 R 03
CIR_CNT_BADV	= 00000002			NICE_ROUTINE	00000324 R 05
CNTRLCMSG	0000004C	R	02	NICE_SIZE	= 00000005
CNTR_TBL	000001A3	R	02	NMASC_CTCIR_ACL	= 0000322
CONTROL_CM	= 00000002			NMASC_CTCIR_CRL	= 0000325
CONTROL_CV	= 00000001			NMASC_CTCIR_DEI	= 00003FC
COUNTER	000001D7	R	03	NMASC_CTCIR_DEO	= 00003FD
COUNTER_MSG	0000011C	R	02	NMASC_CTCIR_IFL	= 0000335
DCS_TERM	***** X		05	NMASC_CTCIR_LBE	= 0000411
DEV	00000048	R	03	NMASC_CTCIR_LDN	= 0000334
DEVBUF	00000053	R	03	NMASC_CTCIR_LIR	= 00004D8
DIBSB_DEVCLASS	= 00000004			NMASC_CTCIR_LPE	= 000044D
DIBSK_LENGTH	= 00000074			NMASC_CTCIR_LRT	= 0000407
END_ADR	000001E7	R	03	NMASC_CTCIR_NIR	= 00004DA
ERROR_COUNT	0000015B	R	03	NMASC_CTCIR_RBE	= 0000410
ERROR_EXIT	000006B3	R	05	NMASC_CTCIR_RIR	= 00004D9
ERR_MSG_CTR	000000E3	R	02	NMASC_CTCIR_RPE	= 000044C
EXIT_DESC	C0000167	R	03	NMASC_CTCIR_RRT	= 0000406
EXIT_HANDLER	0000071C	R	05	NMASC_CTCIR_SLT	= 000041B
FAB\$B_BID	= 00000000			NMASC_CTCIR_TCL	= 000032C
FAB\$B_FNS	= 00000034			NMASC_CTNOD_APL	= 0000384
FAB\$C_BID	= 00000003			NMASC_CTNOD_NOL	= 0000386
FAB\$C_BLN	= 00000050			NMASC_CTNOD_NUL	= 0000385
FAB\$C_SEQ	= 00000000			NMASC_CTNOD_CFL	= 0000387
FAB\$C_VAR	= 00000002			NMASC_CTNOD_PFE	= 000038E
FAB\$L_ALQ	= 00000010			NMASC_CTNOD_RSE	= 0000280

NMASC_CTNOD_RTO	= 00000276	SHRS_ENDEDD	= 00001080
NMASC_CTNOD_RUL	= 00000398	SHRS_OPENIN	= 00001098
NMASC_CTNOD_VER	= 000003A2	SHRS_TEXT	= 00001130
NMASC_ENT_CIR	= 00000003	SHRT_RPRTM	= 00000001
NMASC_ENT_NOD	= 00000000	SHRT_RPRTV	= 00000000
NMASC_FNC_REA	= 00000014	SSS_BADPARAM	***** X 05
NMASC_OPINF_COU	= 00000003	SSS_CONTROLC	***** X 05
NMASC_STS_DDN	= FFFFFF80	SSS_NORMAL	***** X 05
NMASC_STS_MOR	= 00000002	SSS_NOTRAN	***** X 05
NMASC_STS_SUC	= 00000001	SSS_SSFAIL	***** X 05
NMASS_CNT_TYP	= 0000000C	SSS_WASSET	***** X 05
NMASS_CNT_WID	= 00000002	SSERROR	000004F7 R 05
NMASV_CNT_MAP	= 0000000C	STATUS	0000015F R 03
NMASV_CNT_TYP	= 00000000	STSSK_ERROR	= 00000002
NMASV_CNT_WID	= 0000000D	STSSK_INFO	= 00000003
NMASV_OPT_INF	= 00000004	STSSK_SUCCESS	= 00000001
NMLSINITIALIZE	***** X 05	STSSK_WARNING	= 00000000
NMLSPROCESS_NICE	***** X 05	STSSM_INHIB_MSG	= 10000000
NML\$TERMINATE	***** X 05	STSSS_FAC_NO	= 0000000C
NMLINIT_ERR	000000A8 R 02	STSSS_SEVERITY	= 00000003
NOD	00000000	STSSV_FAC_NO	= 00000010
NODE	0000013E R 02	STSSV_SEVERITY	= 00000000
NODE_ADR	00000189 R 03	SUC_EXIT	000002F4 R 05
NODE_ADR_DESC	00000183 R 03	SYSSASSIGN	***** GX 05
NODE_NAME	000001AF R 03	SYSSCLOSE	***** GX 05
NODE_WRD	000001A1 R 03	SYSSCONNECT	***** GX 05
NOD_CNT_BADM	= 00000008	SYSSCREATE	***** GX 05
NOD_CNT_BADV	= 00000003	SYSSDCLEXH	***** GX 05
NO_RMS_AST_TABLE	00000020 R 02	SYSSDISCONNECT	***** GX 05
NRAT_LENGTH	= 00000014	SYSSEXIT	***** GX 05
OTSSCVT_T1_L	***** X 05	SYSSFAO	***** X 05
PC1...	= 0000023F R 02	SYSSGET	***** GX 05
PC2...	= 00000481 R 02	SYSSGETDEV	***** GX 05
RABSB_RAC	= 0000001E	SYSSGETMSG	***** GX 05
RABSC_BID	= 00000001	SYSSOPEN	***** GX 05
RABSC_BLN	= 00000044	SYSSPUT	***** GX 05
RABSC_SEQ	= 00000000	SYSSPUTMSG	***** GX 05
RABSL_CTX	= 00000018	SYSSQIOW	***** GX 05
RABSL_FAB	= 0000003C	SYSSSETAST	***** GX 05
RABSL_RBF	= 00000028	SYSSSETPRN	***** GX 05
RABSL_ROP	= 00000004	SYSSSETFM	***** GX 05
RABSL_STS	= 00000008	SYSTRNLOG	***** GX 05
RABSL_STV	= 0000000C	TBL_END	0000023F R 02
RABSW_RSZ	= 00000022	TBL_SIZE	= 0000001A
RECORD	00000079 R 02	TEXT_BUFFER	= 00000084
RMSSK_FACILITY	= 00000001	THRU	0000014F R 02
RMSS_BLN	***** X 02	TO	0000014C R 02
RMSS_BUSY	***** X 02	TSTNAM	0000000F R 02
RMSS_CDA	***** X 02	TTCHAN	00000000 R 03
RMSS_FAB	***** X 02	TTNAME	0000000A R 03
RMSS_FACILITY	***** X 05	TTNAME_LEN	= 0000000B
RMSS_RAB	***** X 02	TTNAME_ROPTR	00000040 R 02
RMS_COMMON	0000063E R 05	TTNAME_RWPTR	000000C02 R 03
RMS_ERROR	00000608 R 05	TYPE	000001DB R 03
RMS_ERR_STRING	00000087 R 02	TYPE1	000001DF R 03
SHRS_ABENDD	= 000010E0	TYPE2	000001E3 R 03
SHRS_BEGIND	= 00001038	UETNETS00	00000000 RG 05

```

UETP          = 00740000
UETPS_ABENDD = 007410E0
UETPS_ABORTC = 0074832B
UETPS_BEGIND = 00741038
UETPS_ENDEDD = 00741080
UETPS_ERBOXPROC = 00748C20
UETPS_FACILITY = 00000074
UETPS_OPENIN = 00741098
UETPS_TEXT = 00741130
ZERO          = 0000019F R 02

```

```

+-----+
! Psect synopsis !
+-----+

```

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$ABSS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
RODATA	00000481 (1153.)	02 (2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC PAGE
RWDATA	00000314 (788.)	03 (3.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC PAGE
SRMSNAM	0000001A (26.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
UETNETS00	00000768 (1896.)	05 (5.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC PAGE

```

+-----+
! Performance indicators !
+-----+

```

Phase	Page faults	CPU Time	Elapsed Time
Initialization	27	00:00:00.12	00:00:00.92
Command processing	136	00:00:00.72	00:00:03.26
Pass 1	490	00:00:18.77	00:00:40.96
Symbol table sort	0	00:00:02.12	00:00:03.31
Pass 2	204	00:00:04.31	00:00:08.88
Symbol table output	29	00:00:00.22	00:00:00.55
Psect synopsis output	2	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	890	00:00:26.29	00:00:57.91

The working set limit was 2000 pages.

103700 bytes (203 pages) of virtual memory were used to buffer the intermediate code.

There were 90 pages of symbol table space allocated to hold 1519 non-local and 41 local symbols.

1095 source lines were read in Pass 1, producing 34 object records in Pass 2.

50 pages of virtual memory were used to define 43 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name

Macro library name	Macros defined
\$255\$DUA28:[SHRLIB]NMALIBR.Y.MLB;1	1
\$255\$DUA28:[UETP.OBJ]UETP.MLB;1	1
\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	0
\$255\$DUA28:[SYSLIB]STARLET.MLB;2	37
TOTALS (all libraries)	39

1809 GETS were required to define 39 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:UETNETS00/OBJ=OBJ\$:UETNETS00 MSRC\$:UETNETS00/UPDATE=(ENHS:UETNETS00)+EXECMLS/LIB+LIB\$:UETP/LIB

0411 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

UETFORT03
LIS

UETFORT01
LIS

UETPAK00
LIS

UETNETS00
LIS

UETDR1W00
LIS

UETFORT02
LIS